Closing JDBC Objects Explicitly Can Save You Headaches!

Most Java programmers close a connection with the database directly—without closing the *ResultSet* or *Statement*.

For example, take a look at the following code snippet:

```
try
{
    Class.forName( "sun.jdbc.odbc.JdbcOdbcDriver" );

    Connection conn = DriverManager.getConnection("jdbc:odbc:mydb","x","x");
    Statement stmt  = conn.createStatement();
    ResultSet rs    = stmt.executeQuery( "SELECT * FROM TABLE" );
}
catch( SQLException e )
{
    out.println("Some error happened", e);
}
```

This code looks perfectly fine. But not all JDBC drivers clean themselves up.

To be safe, remember to close the everything explicitly in reverse order. Most people do it as shown below:

```
Connection conn = null;
ResultSet rs    = null;
Statement stmt  = null;

try
{
    Class.forName( "sun.jdbc.odbc.JdbcOdbcDriver" );

    conn = DriverManager.getConnection("jdbc:odbc:issue","x","x");
    stmt = conn.createStatement();
    rs   = stmt.executeQuery( "SELECT * FROM TABLE" );
}
catch( SQLException e )
{
    Out.println("Some error happened", e);
}
finally
{
    try
    {
        if (rs  != null)
            rs.close();
        if (stmt  != null)
            stmt.close();
        if (conn  != null)
            conn.close();
    }
    catch (SQLException e)
    {
        // handle exception
    }
}
```